

Package ‘transferegovr’

September 28, 2026

Title Access the 'TransfereGov' Open Data APIs

Version 0.2.0

Description Provides a modern interface to the open data application programming interfaces of the Brazilian federal government's 'TransfereGov' platform ([\(<https://www.gov.br/transferegov/pt-br/ferramentas-gestao/dados-abertos>\)](https://www.gov.br/transferegov/pt-br/ferramentas-gestao/dados-abertos)). Covers the special transfers, fund-to-fund transfers, partnership management, and decentralized credit ('TED') modules, which together publish seventy-four tables on action plans, programs, proposals, partnerships, budget commitments, credit notes, financial execution, management reports, and payment orders. Filters are the services' own typed query parameters, validated against the published schema before a request is made, and results are returned as tidy tibbles with types taken from that schema. Automatic pagination, request throttling, retries with exponential backoff, and an optional response cache are included.

License MIT + file LICENSE

URL <https://github.com/StrategicProjects/transferegovr>,
<https://strategicprojects.github.io/transferegovr/>

BugReports <https://github.com/StrategicProjects/transferegovr/issues>

Depends R (>= 4.1.0)

Imports cli, httr2 (>= 1.0.0), purrr (>= 1.0.0), rlang (>= 1.1.0),
stats, tibble (>= 3.2.0), utils

Suggests covr, dplyr, jsonlite, knitr, rmarkdown, testthat (>= 3.2.0),
tidyr, withr

VignetteBuilder knitr

Config/Needs/website pkgdown

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Andre Leite [aut, cre] (ORCID: <<https://orcid.org/0000-0002-4718-9766>>),
Marcos Wasiliew [aut],
Hugo Vasconcelos [aut] (ORCID: <<https://orcid.org/0000-0001-6249-0920>>),
Carlos Amorim [aut] (ORCID: <<https://orcid.org/0000-0001-6315-8305>>),
Diogo Bezerra [aut] (ORCID: <<https://orcid.org/0000-0002-1216-8674>>),
Júlia Nascimento Barreto [aut]

Maintainer Andre Leite <leite@castlab.org>

Repository CRAN

Date/Publication 2026-09-28 17:00:02 UTC

Contents

module_shortcuts	2
tg_base_url	3
tg_cache_clear	4
tg_cache_dir	4
tg_count	5
tg_fields	6
tg_get	7
tg_metadata	9
tg_modules	10
tg_params	10
tg_schema_date	11
tg_tables	12
tg_updated_at	13
Index	14

module_shortcuts	<i>Query a single module</i>
------------------	------------------------------

Description

Thin wrappers over `tg_get()` with the module fixed, for code that stays within one API.

Usage

```
tg_parcerias(table, ...)  
  
tg_fundo_a_fundo(table, ...)  
  
tg_especiais(table, ...)  
  
tg_ted(table, ...)
```

Arguments

`table` A table name from `tg_tables()` for that module.

`...` Passed to `tg_get()`: filters, and any of its `.`-prefixed arguments.

Value

A tibble, as `tg_get()` returns.

See Also

Other queries: `tg_count()`, `tg_get()`, `tg_metadata()`

Examples

```
if (interactive()) {
  tg_parcerias("proposta", .limit = 10)
  tg_fundo_a_fundo("programas", .limit = 10)
  tg_especiais("programas_especiais", .limit = 10)
  tg_ted("termos_execucao", .limit = 10)
}
```

tg_base_url	<i>The API base URL in use</i>
-------------	--------------------------------

Description

Reports the base URL requests are sent to. Set the `transferegovr.base_url` option to point the package at a mirror or a test double.

Usage

```
tg_base_url()
```

Value

A single string.

Examples

```
tg_base_url()
```

tg_cache_clear	<i>Delete cached responses</i>
----------------	--------------------------------

Description

Delete cached responses

Usage

```
tg_cache_clear()
```

```
tg_cache_limpar()
```

Value

The number of files removed, invisibly.

See Also

Other cache: [tg_cache_dir\(\)](#)

Examples

```
tg_cache_clear()
```

tg_cache_dir	<i>Where cached responses are stored</i>
--------------	--

Description

Called with no argument, reports the directory in use. Called with a path, switches to it for the rest of the session.

Usage

```
tg_cache_dir(path = NULL)
```

```
tg_cache_pasta(path = NULL)
```

Arguments

path	A directory to cache responses in, or NULL to report the current one. The directory is created if it does not exist.
------	--

Details

By default responses are cached in the session’s temporary directory, so they are discarded when R exits. To keep them between sessions, set this to a persistent path, for example `tg_cache_dir(tools::R_user_dir("transferegovr", "cache"))`, or set the `TRANSFEREGOVR_CACHE_DIR` environment variable in your `.Renv` file.

Caching is controlled by the `transferegovr.cache` option (`TRUE` by default) and entries expire after `transferegovr.cache_ttl` seconds (3600 by default). The data behind these APIs is refreshed daily.

Value

The cache directory, invisibly when setting it.

See Also

Other cache: `tg_cache_clear()`

Examples

```
tg_cache_dir()
```

tg_count	Count the rows a query matches
----------	--------------------------------

Description

Asks the API for the number of rows matching a set of filters without retrieving them. Worth doing before a large `tg_get()`: the biggest table in these APIs holds over a million rows, which at 200 rows a request is more than five thousand requests.

Usage

```
tg_count(module, table, ..., .cache = NULL, .base_url = NULL)

tg_contar(module, table, ..., .cache = NULL, .base_url = NULL)
```

Arguments

- `module` A module name from `tg_modules()`: "especiais", "fundoaofundo", "parcerias" or "ted". Aliases such as "fundo_a_fundo" are accepted.
- `table` A table name from `tg_tables()`.
- `...` Filters, named after the parameters they set. See `tg_get()`.
- `.cache` Whether to serve the request from the response cache. `NULL` follows the `transferegovr.cache` option. See `tg_cache_dir()`.
- `.base_url` The API base URL. Defaults to `tg_base_url()`.

Value

A single number.

See Also

Other queries: [module_shortcuts](#), [tg_get\(\)](#), [tg_metadata\(\)](#)

Examples

```
if (interactive()) {
  tg_count("parcerias", "proposta")
  tg_count("parcerias", "proposta", situacao_proposta = "Aprovada")
}
```

tg_fields	<i>List the columns of a table</i>
-----------	------------------------------------

Description

Column names stay in Portuguese because they are the API's own contract. Not every column can be filtered on; [tg_params\(\)](#) lists the ones that can.

Usage

```
tg_fields(module, table, nested = NULL)
```

```
tg_campos(modulo, tabela, nested = NULL)
```

Arguments

module	A module name from tg_modules() . Aliases such as "funcao_a_funcao" are accepted. NULL lists the tables of every module.
table	A table name from tg_tables() .
nested	The name of a list column, to describe the columns of the objects inside it instead of the table's own. NULL, the default, describes the table.
modulo	Portuguese alias for module, available in tg_tabelas() and tg_campos() .
tabela	Portuguese alias for table, available only in tg_campos() .

Value

A tibble with one row per column: its name, the R type the package coerces it to, the type the API declares, the sub-schema it nests when it is a list column, and its description.

See Also

Other discovery: [tg_modules\(\)](#), [tg_params\(\)](#), [tg_schema_date\(\)](#), [tg_tables\(\)](#), [tg_updated_at\(\)](#)

Examples

```
tg_fields("parcerias", "proposta")

# A list column, and what it holds
fields <- tg_fields("parcerias", "proposta")
fields[!is.na(fields$nested), c("field", "nested")]
tg_fields("parcerias", "proposta", nested = "intervenientes_proposta")
```

tg_get

Retrieve rows from a TransfereGov table

Description

Queries one of the seventy-four tables published by the TransfereGov open data APIs and returns them as a tibble, with columns typed from the API's own schema.

Usage

```
tg_get(
  module,
  table,
  ...,
  .limit = 1000,
  .offset = 0,
  .page_size = NULL,
  .progress = NULL,
  .cache = NULL,
  .base_url = NULL
)

tg_obter(
  module,
  table,
  ...,
  .limit = 1000,
  .offset = 0,
  .page_size = NULL,
  .progress = NULL,
  .cache = NULL,
  .base_url = NULL
)
```

Arguments

module	A module name from <code>tg_modules()</code> : "especiais", "fundoafundo", "parcerias" or "ted". Aliases such as "fundo_a_fundo" are accepted.
table	A table name from <code>tg_tables()</code> .

...	Filters, named after the parameters they set. See the Filters section.
.limit	Maximum number of rows to return. Use Inf for every matching row.
.offset	Number of matching rows to skip before the first one returned.
.page_size	Rows per request. NULL, the default, asks for the largest page the module serves, which <code>tg_modules()</code> reports as <code>max_page_size</code> .
.progress	Whether to show a progress bar while collecting pages. NULL shows one in interactive sessions when more than one page is needed.
.cache	Whether to serve the request from the response cache. NULL follows the <code>transferegovr.cache</code> option. See <code>tg_cache_dir()</code> .
.base_url	The API base URL. Defaults to <code>tg_base_url()</code> .

Value

A tibble. `tg_metadata()` reports the totals the API gave and how many pages were fetched. A column the API sends as an array of objects comes back as a list column; `tg_fields()` describes what is inside it.

Filters

Name each filter after one of the table's query parameters and give it a value. Parameters are combined with AND:

```
tg_get("parcerias", "proposta", situacao_proposta = "Aprovada")
tg_get(
  "parcerias", "proposta",
  sg_uf_recebedor = "PE", ano_proposta = 2025
)
```

The services compare for equality: there is no greater-than and no pattern match. Most parameters take one value. Some identifier parameters take several and match any of them — `tg_params()` marks them as `multiple`, with the most each accepts in `max_values`:

```
tg_get("ted", "planos_acao metas", id_plano_acao = c(3, 4))
```

For any other parameter, query each value and bind the results.

Parameter names, and the permitted values of the enumerated ones, are in Portuguese because they belong to the API. Use `tg_params()` to see them. A name the packaged schema does not know is an error rather than a request: these services ignore a parameter they do not recognize and answer with the whole table, so an unchecked typo would return plausible, wrong data.

Pagination

Each request returns one page of at most the module's page limit — 200 rows for `especiais` and `parcerias`, 1000 for `fundoafundo` and `ted` — so a larger `.limit` is met by fetching successive pages. `.limit` counts rows, not pages; use Inf for every matching row. Several tables hold hundreds of thousands of rows, so check the size with `tg_count()` first.

Row order is the server's and cannot be set: these APIs publish no ordering parameter. It was checked to be stable across page sizes, across repeated calls and at depth, which is what makes multi-page collection safe. The number of rows collected is checked against the total the API reports, and a mismatch is reported as a warning.

See Also

Other queries: [module_shortcuts](#), [tg_count\(\)](#), [tg_metadata\(\)](#)

Examples

```
if (interactive()) {
  tg_get("parcerias", "proposta", sg_uf_recebedor = "PE", .limit = 50)

  tg_get("fundoafundo", "planos_acao", .limit = 10)
}
```

tg_metadata

Inspect what a query retrieved

Description

Inspect what a query retrieved

Usage

```
tg_metadata(x)

tg_metadados(x)
```

Arguments

x A tibble returned by [tg_get\(\)](#).

Value

A list holding the module and table queried, the total number of matching rows the API reported, how many rows and pages were retrieved, the offset, page size and filters used, and when the query ran. NULL for any other object.

See Also

Other queries: [module_shortcuts](#), [tg_count\(\)](#), [tg_get\(\)](#)

Examples

```
tg_metadata(tibble::tibble())
```

tg_modules	<i>List the TransfereGov API modules</i>
------------	--

Description

List the TransfereGov API modules

Usage

```
tg_modules()
```

```
tg_modulos()
```

Value

A tibble with one row per module: its name, the label used in this documentation, the number of tables it publishes, the largest page it serves in one request, and its API base URL.

See Also

Other discovery: [tg_fields\(\)](#), [tg_params\(\)](#), [tg_schema_date\(\)](#), [tg_tables\(\)](#), [tg_updated_at\(\)](#)

Examples

```
tg_modules()
```

tg_params	<i>List the parameters a table accepts as filters</i>
-----------	---

Description

Every parameter may be passed to [tg_get\(\)](#) and [tg_count\(\)](#) as a named argument. Parameter names and their permitted values are in Portuguese because they belong to the API.

Usage

```
tg_params(module, table)
```

```
tg_parametros(modulo, tabela)
```

Arguments

module	A module name from tg_modules() . Aliases such as "funcao_a_funcao" are accepted. NULL lists the tables of every module.
table	A table name from tg_tables() .
modulo	Portuguese alias for module, available in tg_tabelas() and tg_campos() .
tabela	Portuguese alias for table, available only in tg_campos() .

Value

A tibble with one row per parameter: its name, the R type a value should have, the type the API declares, the permitted values when the parameter is enumerated, the pattern a value must match when it has one, its description, whether it accepts several values (`multiple`), and how many at most (`max_values`).

See Also

Other discovery: [tg_fields\(\)](#), [tg_modules\(\)](#), [tg_schema_date\(\)](#), [tg_tables\(\)](#), [tg_updated_at\(\)](#)

Examples

```
tg_params("parcerias", "proposta")

# Which parameters accept only a fixed set of values?
params <- tg_params("parcerias", "proposta")
params[lengths(params$values) > 0, c("param", "values")]

# Which accept several values at once?
params[params$multiple, c("param", "max_values")]
```

`tg_schema_date`*Report the frozen schema's build date*

Description

The package validates filters and types columns against a copy of the APIs' OpenAPI documents taken on this date. A column added upstream since then is still returned, but is typed by inspection rather than from the schema.

Usage

```
tg_schema_date()
```

Value

A Date.

See Also

Other discovery: [tg_fields\(\)](#), [tg_modules\(\)](#), [tg_params\(\)](#), [tg_tables\(\)](#), [tg_updated_at\(\)](#)

Examples

```
tg_schema_date()
```

tg_tables

List the tables a module publishes

Description

List the tables a module publishes

Usage

```
tg_tables(module = NULL, counts = FALSE)
```

```
tg_tabelas(modulo = NULL, contagens = FALSE)
```

Arguments

module	A module name from tg_modules() . Aliases such as "fundo_a_fundo" are accepted. NULL lists the tables of every module.
counts	If TRUE, adds a rows column with the number of rows each table currently holds. This is the only part of this function that needs a network connection: it makes one request per table, so <code>tg_tables(counts = TRUE)</code> with no module makes seventy-four. Responses are cached.
modulo	Portuguese alias for module, available in tg_tabelas() and tg_campos() .
contagens	Portuguese alias for counts, available only in tg_tabelas() .

Value

A tibble with one row per table: its module, name, the endpoint path it maps to, its number of columns and filterable parameters, and the description published in the API schema. With `counts = TRUE`, also the current number of rows.

See Also

Other discovery: [tg_fields\(\)](#), [tg_modules\(\)](#), [tg_params\(\)](#), [tg_schema_date\(\)](#), [tg_updated_at\(\)](#)

Examples

```
tg_tables("parcerias")
tg_tables()

if (interactive()) {
  # How big is everything, largest first?
  sizes <- tg_tables(counts = TRUE)
  sizes[order(-sizes$rows), ]
}
```

tg_updated_at	<i>When a module's data was last refreshed</i>
---------------	--

Description

Each module publishes the timestamp of its last load. It is the only freshness signal these APIs give: they send no ETag, Cache-Control or Last-Modified header.

Usage

```
tg_updated_at(module, .cache = NULL, .base_url = NULL)
```

```
tg_atualizado_em(module, .cache = NULL, .base_url = NULL)
```

Arguments

module	A module name from tg_modules() : "especiais", "fundoafundo", "parcerias" or "ted". Aliases such as "fundo_a_fundo" are accepted.
.cache	Whether to serve the request from the response cache. NULL follows the <code>transferegovr.cache</code> option. See tg_cache_dir() .
.base_url	The API base URL. Defaults to tg_base_url() .

Value

A POSIXct in UTC.

See Also

Other discovery: [tg_fields\(\)](#), [tg_modules\(\)](#), [tg_params\(\)](#), [tg_schema_date\(\)](#), [tg_tables\(\)](#)

Examples

```
if (interactive()) {  
  tg_updated_at("parcerias")  
}
```

Index

- * **cache**
 - tg_cache_clear, 4
 - tg_cache_dir, 4
- * **configuration**
 - tg_base_url, 3
- * **discovery**
 - tg_fields, 6
 - tg_modules, 10
 - tg_params, 10
 - tg_schema_date, 11
 - tg_tables, 12
 - tg_updated_at, 13
- * **queries**
 - module_shortcuts, 2
 - tg_count, 5
 - tg_get, 7
 - tg_metadata, 9

module_shortcuts, 2, 6, 9

tg_atualizado_em (tg_updated_at), 13

tg_base_url, 3

tg_base_url(), 5, 8, 13

tg_cache_clear, 4

tg_cache_clear(), 5

tg_cache_dir, 4

tg_cache_dir(), 4, 5, 8, 13

tg_cache_limpar (tg_cache_clear), 4

tg_cache_pasta (tg_cache_dir), 4

tg_campos (tg_fields), 6

tg_campos(), 6, 10, 12

tg_contar (tg_count), 5

tg_count, 5

tg_count(), 3, 8–10

tg_especiais (module_shortcuts), 2

tg_fields, 6

tg_fields(), 8, 10–13

tg_fundo_a_fundo (module_shortcuts), 2

tg_get, 7

tg_get(), 2, 3, 5, 6, 9, 10

tg_metadados (tg_metadata), 9

tg_metadata, 9

tg_metadata(), 3, 6, 8, 9

tg_modules, 10

tg_modules(), 5–8, 10–13

tg_modulos (tg_modules), 10

tg_obter (tg_get), 7

tg_parametros (tg_params), 10

tg_params, 10

tg_params(), 6, 8, 10–13

tg_parcerias (module_shortcuts), 2

tg_schema_date, 11

tg_schema_date(), 6, 10–13

tg_tabelas (tg_tables), 12

tg_tabelas(), 6, 10, 12

tg_tables, 12

tg_tables(), 3, 5–7, 10, 11, 13

tg_ted (module_shortcuts), 2

tg_updated_at, 13

tg_updated_at(), 6, 10–12